PPRC Research Group

Scott McFarling

PPRC-RES

Mission:

Advance PPRC programmer productivity, testing, and performance goals by:

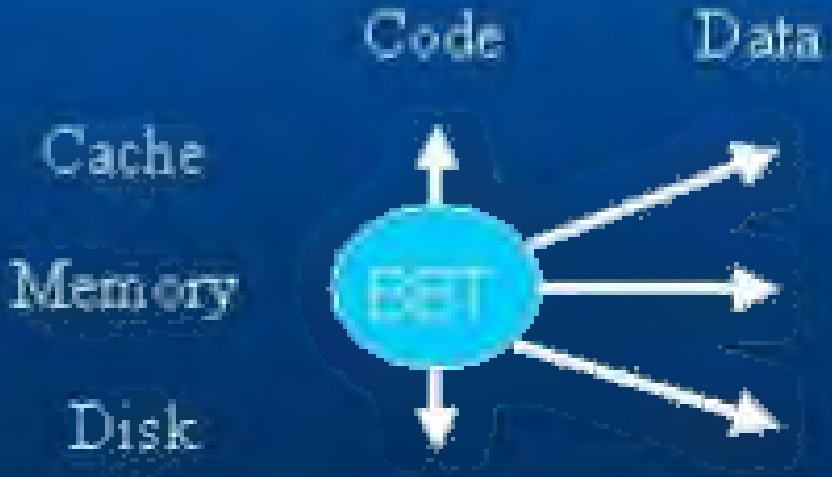
- Creating New Core Technology
- Seeding New Projects
- Collaborating with other Groups

Current Projects

- Memory Optimization
  - Better Profiling
  - Malloc
- Testing
  - Coverage Enhancement
  - Test Prioritization
- Tools
  - Binary Matching
  - Linker

BBT2000 Overview

- Better Optimization with Better Information
- Make Better Use of Existing Information
- Optimize OS Interaction
- Data
  - Static
  - Heap



BBT2000: Profiling Overview

- Problems with Current Profiling
  - 20X slowdown
    - Speed Effects Behavior
    - Can't instrument all NT dll's
  - Limited Scenarios
    - Short Build Window
    - Large Complex Highly Variable Applications
    - Each Untrained Block: Potential Expensive Seek
- Solution:
  - 2000X "faster" profiling
  - 2000X more scenario data
  - Toward Real Use as the Scenario

Step 1: Tuning Instrumentation

- 10X faster
- 10X more scenarios can be run
- Instrument all NT dlls

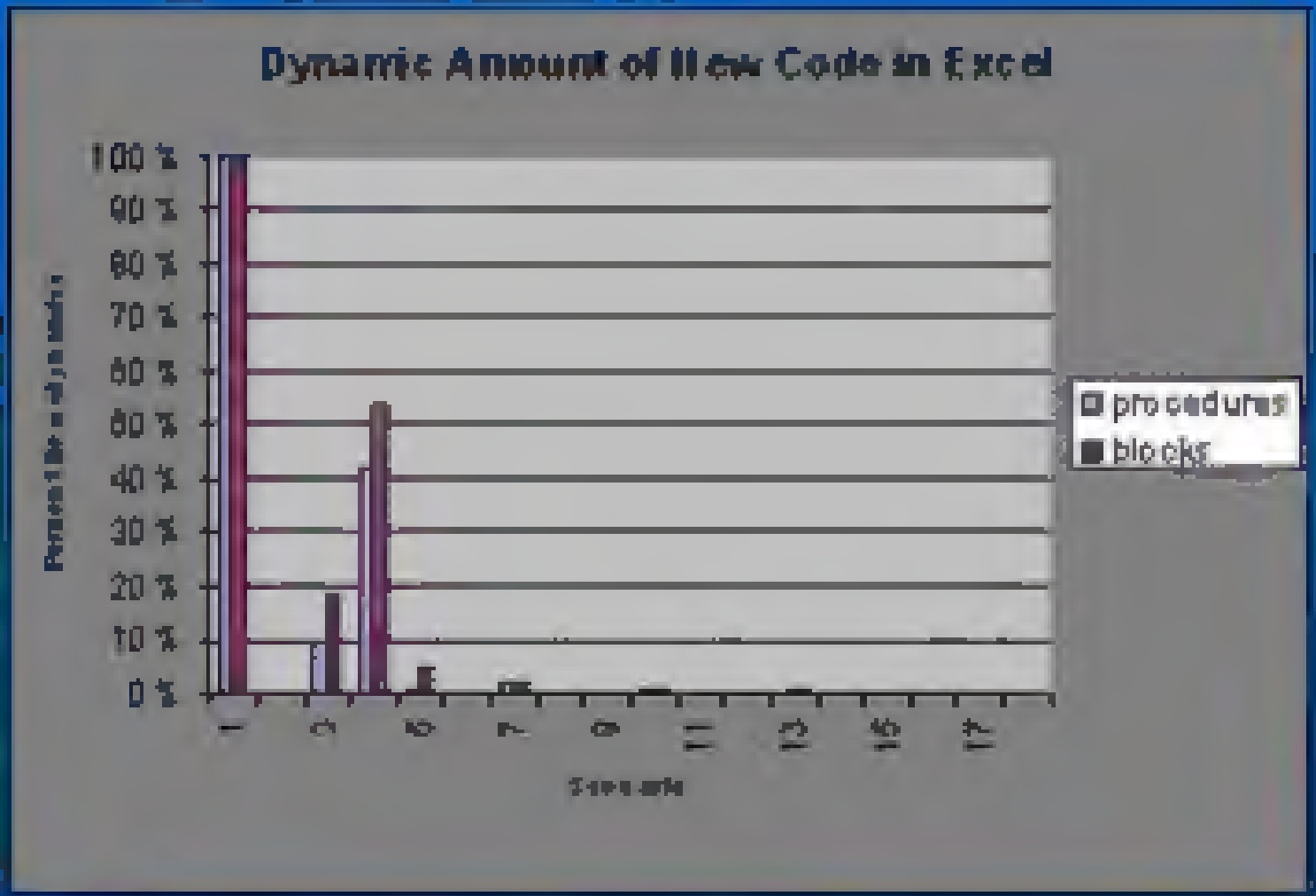
Step 2: Background Profiling

- Scenarios run round-robin continuously
  - Ex. 2 week cycle (336 hrs / 1.5 hour now) = 200X
  - Avg. 1 week stale
- Need effective use of stale profiles
  - Backup Phase
  - Doesn't disturb prior gains
  - Can tolerate some match problems
- Goals
  - Within 1% of full runs
  - Pure win over build time data
  - Need high quality binary matching
- Next Problem: may run out of scenarios

Step 3: Incremental profiling

- Profile the Unprofiled
- Second Phase: no need to collect if covered
- Goals:
  - 10X lower overhead
  - 10% slowdown
  - 2000X more scenarios?
  - Enable Real User Profiling

Incremental Profiling Overhead

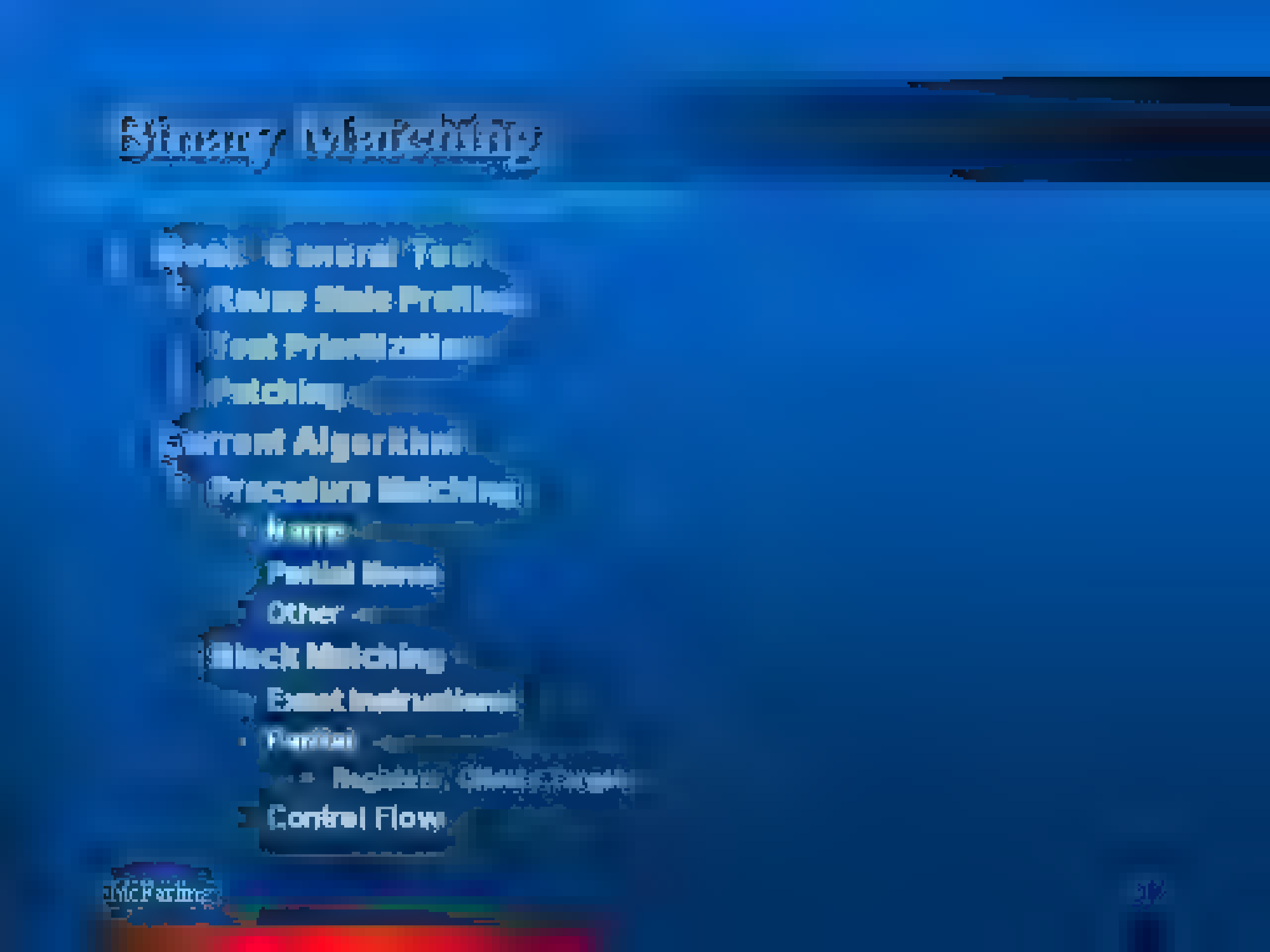


Alternative: Massively Parallel Profiling

- What kinds of information can we collect from very large numbers of users?
  - Need low overhead, uninstrumented binaries, infrastructure for reporting and analysis
- First idea: profiling
  - Interrupt at some rate; log various performance counters; flush logs to network
  - Example: bottlenecks under real workloads
  - Example: does cache behavior deteriorate with time?

Step 4: Refine Global Profiles

- Need way to deal with 2000X data
- Scalable Locality Model
  - FOR granularity
  - Loss may not be appropriate for all cases
- Re-examine Affinity Models
  - Closer Match to Real Cost
  - Cluster/Page/Cache
- Server App Issues


Control Flow

Extended Data Table 1 | Model performance

Metrics

Cross-validation

Hold-out

Fold	Acc	Sn	Sp	Auc
Hold-out	?	?	?%	?%
Universal	99.97%	99.59%	99.81%	99.85%
Contact 3Å	99.92%	96.77%	99.76%	93.63%
Interface	99.50%	100.00%	99.74%	99.19%
Explorer	99.97%	97.08%	99.74%	95.18%
Metagen	99.92%	99.83%	99.91%	99.40%
Indoor	99.95%	99.49%	99.28%	99.55%
Shell3Å	99.98%	99.16%	99.61%	97.00%
Shield	99.99%	99.99%	99.97%	97.16%
Average	99.85%	99.33%	99.16%	97.97%

Data Profiling

Place Interesting Objects on Separate Pages

Monitor Use Through Page Faults

Can Clear in Phases

TLBs for Data

Status

Negligible Overhead

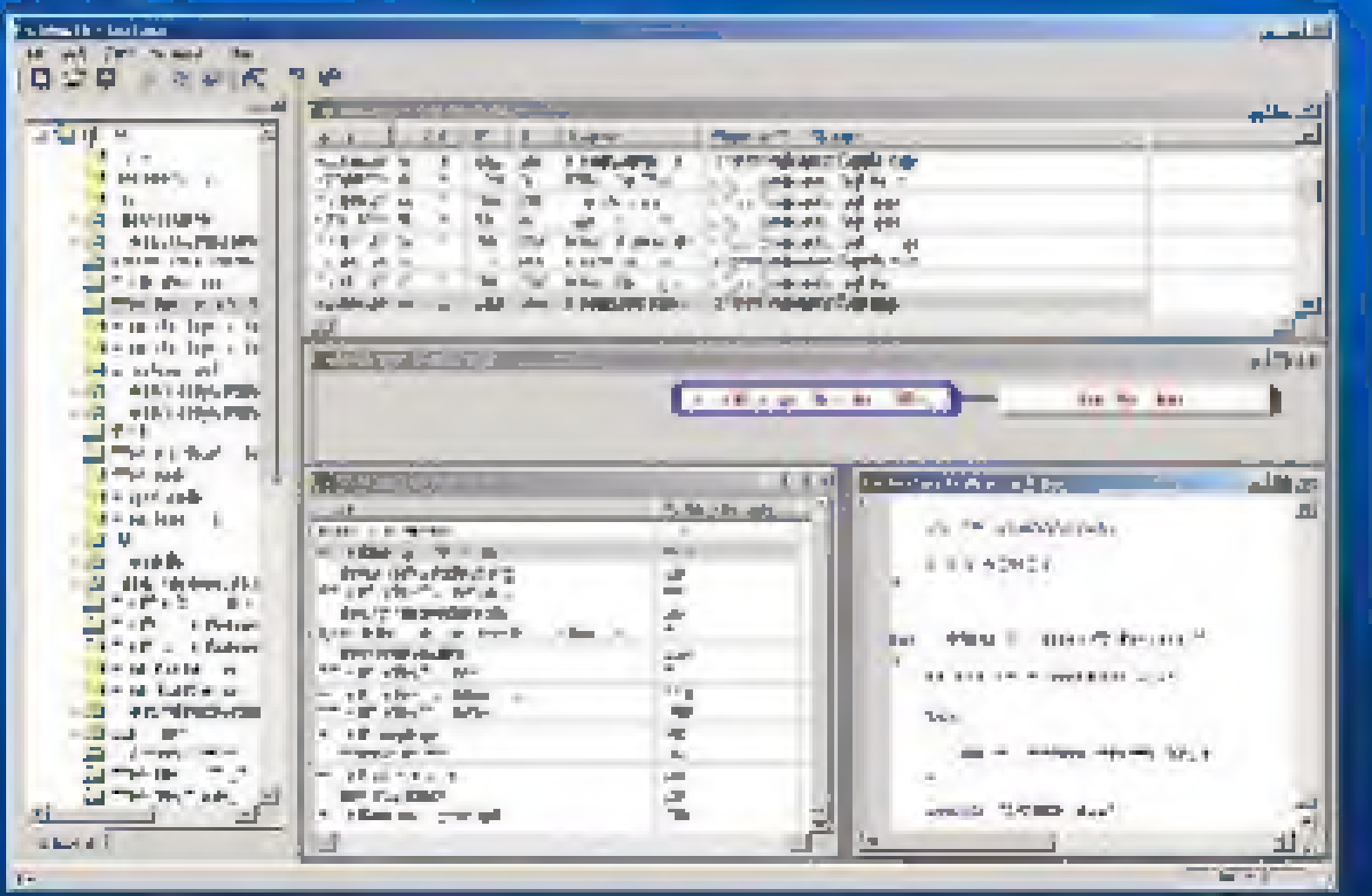
Evaluating Gains for Static Data

17.5% WS for FoxPro

Memory Allocation

- Scale to Multiple Threads
- Fast Allocation
- Customizable
- What Features Required for MS Products?
- Future
  - Profile Driven (BBT for Data)
  - Detect Memory Allocation Errors

Test Coverage Enhancement: Sleuth



Test Prioritization

- Optimize Order Tests Run
  - Test Recent Changes
  - Source Level Changes
  - Binary Level Changes
  - Coverage vs. Speed
- Status: Evaluating Binary Matching

Conclusion

Investigating some new approaches

Starting to have an impact on existing tools